Utilizing External Information in Social Networks

David J. Marchette*

Abstract

External information about a social network is of three basic types: information about the actors within the network, such as their roles in society, their personal attributes, and so on; information about the links may be categorical, such as an indicator of the type of link, or may be information about the specific link, such as the content of a communication; finally, there may be environmental information, describing the environment in which the social network resides. We will describe methods for handling some of these types of external information from the point of view of the fusion of disparate data. The social network model, a latent space model, provides a natural mechanism for performing this fusion.

Key Words: Latent space model ; social network analysis ; random dot product graph ; data fusion

1. Introduction

Many (if not most) decision making is performed in an environment in which relationships among the actors are critical to making correct decisions. Methods are needed for performing inference on these types of relationship data. Considerable work has been done on these problems in the social network analysis community (see for example Wasserman and Faust [1994], Carrington et al. [2005]). Relationship data can be augmented with information about the actors as well as the relationships, and external information may be available about the environment in which the data live. Thus, statistical methods appropriate to these complex types of data are needed to aid the decision making process. This paper discusses some examples and provides some suggestions for new approaches to inference on complex data sets of this type.

1.1 Graphs and Networks

A graph is a pair $G = (V = [n], E)$, where the set V corresponds to the vertices or nodes of the graph and E, the edge set, is set of unordered pairs of vertices (if the graph is directed, these are ordered pairs). In social network analysis the vertices are often called *actors* and the edges *relations* or *relationships*. Throughout this paper we assume the graph is simple, that is, there are no loops (edges from a vertex to itself) and no multiple edges (although either of these conditions can be accommodated).

A network is usually defined as a directed graph with weighted edges, but we will use a broader definition. We define a network to be a graph (directed or undirected) with weights (scalars or vectors, or even more complicated data structures) on either the vertices, the edges, or both.

Graphs and networks are used to describe many data sets and problems of importance to policy makers, including:

*Naval Surface Warfare Center, 18444 Frontage Rd, Suite 327, Dahlgren, VA 22448

- Nation states and their relationships such as alliances, trade, conflict, et cetera.
  - Which nations are politically stable, and which are unstable?
  - Is conflict likely between nations A and B?
  - What would be the result (on some subset of nations) of taking a particular action?
- Computer networks and their logical connections.
  - Are there attacks on the network, and what are their characteristics?
  - How vulnerable is the network?
  - Has the network changed in an important way (such as a new application (file sharing, chat, twitter, are examples from the past).
- Criminal organizations and their communications or other relationships.
  - How effective is the organization at reaching its goals?
  - What are the effects of taking a particular action against the organization?
  - What is the organization likely to do in the future?
- Sensor networks.
  - How should the network be processed to provide optimal detection?
  - Where should resources be spent to improve the overall performance?
  - How should the network be optimally deployed?

These and other problems lead us to the requirement to perform inference on complex data sets containing both information about entities and information about the relationships among entities.

1.2 Inference on Graphs

Inference can take (at least) two distinct forms: inference about the entities in a particular graph (collection of graphs, time series of graphs); or inference about the process that generated the graph(s). In either case, the graph may be modeled as random, and there may be random covariates and edge weights as well. Typical inferences include:

- A vertex or group of vertices is "acting differently" than the others or than it has in the past.
- The graph was generated by this process with these parameters (or not).
- The random graph process has changed.

- Certain vertices (or edges) are "important" (central, bridges, sources/sinks, choke-points).
- Certain structure is/is not present in the graph.
- Information flows most efficiently along these paths, through these vertices.
- The graph does/does not predict the covariates (or vice-versa).

We consider three basic approaches to inference on graphs: feature extraction, modeling, and embedding. The first two could be considered as special cases of the third. There is a rich literature on modeling random graphs, from the basic Erdös-Renyí model (see Bollobás [2001]), to modern models of social networks (see Wasserman and Faust [1994], Hoff et al. [2002], Carrington et al. [2005], Hoff [2005] and Kolaczyk [2009] among others). Instead, we will focus on embedding techniques.

2. Fusing Graph and External Data

The basic setup is as follows. We are given $D_t = (G_t, H_t, Z_t)$, where

- $G_t = (V = [n], E_t)$ is an unweighted graph.
- $H_t = (V, W_t)$ is a weighted graph on the same vertex set V, with edge weights W_t.
- Z_t are covariates on the vertices: Z_t is an $n \times q$ matrix.

Note that we are indexing the data by time, since in many applications the data change in time, and time series methods may be worth considering. We will denote by $\mathcal{D}$ the space defined by the data D_t (we will assume that the overall space $\mathcal{D}$ does not depend on time). This is admittedly a special case of the overall problem, but it has sufficient complexity to illustrate various ideas.

The approach we will take to inference is that of embedding. We will attempt to embed the data into a d dimensional space and perform inference on the embedded points. The main reason for doing this is to take advantage of the myriad methodologies developed for performing inference on vectors in Euclidean space (or other nice metric spaces). Thus we seek to define $h : \mathcal{D} \to \mathbb{R}^d$, and we would like to ensure that h has some set of nice properties with respect to inference. Ideally, given a measure of quality of inference β (such as power of a hypothesis test against a given alternative or set of alternatives, probability of correct classification of a classifier, et cetera), we would like to find:

$$h^* = \arg\max_h \beta(h; \mathcal{D}). \tag{1}$$

In general, this may be intractable, but we will keep this ultimate goal in mind as we proceed. It should be noted that we are considering the embedding to embed the vertices of the graphs (the actors, or nodes) rather than embedding the graph as a single point in the embedded space. This latter would be appropriate for some problems (for example, determining that there has been a change in the distribution underlying the observations), but we will focus on the former.

There are several ways we might choose to proceed in defining the embedding. We can embed each piece of D_t separately:

$$\begin{aligned} G_t &\mapsto X_{G_t} \in \mathbb{R}^{n \times d_G} \\ H_t &\mapsto X_{H_t} \in \mathbb{R}^{n \times d_H} \\ Z_t &\mapsto X_{Z_t} \in \mathbb{R}^{n \times d_Z} \end{aligned}$$

(where, perhaps, $X_{Z_t} = Z_t$), and then combine the embeddings (for example, via Cartesian product)

$$(X_{G_t}, X_{H_t}, X_{Z_t}) \subset \mathbb{R}^{d_G+d_H+d_Z}.$$

Alternatively, given the different embeddings, if we can assume that they are embedded into the same space (and hence $d_G = d_H = d_Z$) and are thus commensurate, we can perform a Procrustes mapping to align the different embeddings of the vertices. One may then obtain an "average" of these embeddings as an overall representation of the vertices. Ensuring that this makes sense requires some work, and we will not address this approach further in this paper.

The best approach, assuming it is practical, would seem to be to perform the embedding in one step:

$$D_t = (G_t, H_t, Z_t) \mapsto \mathbb{R}^{n\times d}.$$

In principle, this should be superior to a method that embeds the pieces individually then seeks to combine the embeddings. One of the contributions of this paper is to discuss some methods for realizing such a simultaneous embedding.

We will consider a spectral method for embedding graph data. This approach is related to random dot product graphs (Marchette and Priebe [2008]), latent position models (Hoff et al. [2002]) and correspondence analysis (Faust [2005]). The idea is to decompose the adjacency matrix (or a modified version of this matrix) into a product of two matrices (essentially the eigenvectors of the matrix). These matrices then define the embedding.

To understand the correspondence analysis version of this, consider a two-mode social network. That is, a bipartite network in which actors from set A have edges only to actors in set B and vice versa. For example, the two sets may be the people on the boards of directors (A) and the associated companies (B), legislators and bills, people and clubs, and so on. One forms M, the incidence matrix, the $n \times m$ binary matrix with a 1 in m_{ij} if and only if there is an edge between a_i and b_j. This matrix is then scaled by matrices S and T (see Faust [2005] for discussion on various scaling schemes and their consequences):

$$M' = SMT' = X\Lambda Y',$$

and the first d columns of X and Y (scaled appropriately by Λ) are the embeddings of the two sets of actors. In the situation we are concerned with, the graph is not bipartite, but the basic approach still holds. We consider two variants. In the first, we augment the diagonal of the adjacency matrices as follows:

$$\begin{aligned} D_G &= \text{diag}(\text{deg}_i/(n_t - 1)) \\ D_H &= \text{diag}(\text{rowSum}_i/(n_t - 1)) \\ B_G &= D_G + A_G \\ B_H &= D_H + A_H \\ A &= (B_G, B_H). \end{aligned}$$

We then use the singular value decomposition to decompose $A = XY'$ and use the first d columns of X as the embedding. This is related to the random dot product model of Marchette and Priebe [2008].

The random dot product model posits a set of vectors $\{x_1, \ldots, x_n\} \subset \mathbb{R}^2$, one per node, so that the probability of an edge between nodes i and j is defined by the dot product of the vectors:

$$P[i \sim j] = f(x_i x_j')$$

for a suitable function f (usually a thresholding function). The above embedding incorporates the edge information in both graphs into the embedding.

Basically, these embedding methods are latent space models: they posit a unobserved (latent) space in which the information appropriate to inference "lives", and provide methods for finding this space. A more explicit representation of social networks in terms of a latent space is presented in Hoff et al. [2002].

Another approach is to form the following matrices:

$$S = \begin{pmatrix} B_G & U' \\ U & B_H \end{pmatrix}$$
$$D = \begin{pmatrix} D_G & W' \\ W & D_H \end{pmatrix}$$

Here, D_G and D_H are dissimilarity matrices defined by the graphs. The matrices U and W need to be defined, and we use the average of the diagonal matrices. Using S the same singular value approach leads to an embedding. Using D, multidimensional scaling is appropriate (see Ma and Priebe [2009]). Note that in this case, each vertex has two embedding points, one corresponding to G and the other to H, but these are commensurable. The distance between pairs can give a measure of how much the two representations are consistent with each other.

A third approach is inspired by the random projection idea (see for example Ailon and Chazelle [2009] and references therein). Assign to each vertex v a vector $U_v \in \mathbb{R}^d$. This vector may be drawn from some given distribution (hence the relationship to random projections), or it may be some function of covariates, or some combination of these two ideas. Associate to each edge $e = vw$ the vector $g(U_v, U_w)$ for a user specified function g. We will use term-by-term multiplication. Let $s(v)$ be a subgraph containing v (we will use the subgraph consisting of the edges incident on v, and their associated vertices). Finally, associate to each v the vector:

$$X_v = \frac{1}{|E_{s(v)}|} \sum_{e \in E_{s(v)}} g(e).$$

Thus, the vector is the centroid of its associated edges (in our case, the centroid of its incident edges). This approach can be easily extended to hypergraphs (see Hohman and Marchette [2009]), and to $A = (B_G, B_H)$ as above.

Incorporating the covariates can be done in several ways. They can be combined using the direct product in any of the embedding methods. They can be incorporated directly into A in the first method:

$$A = (B_G, B_H, Z).$$

They can be used to define the vectors U in the random projection method, either deterministically (conditionally on the value of Z) or in addition to some random component. In the case in which the covariates provide information about the connectivity, utilizing them in the embedding directly should reduce variance. If the information in the covariates is orthogonal to that in the graph, the direct product approach should reduce bias.

An important issue that we have not discussed is scaling. When all information is on the same scale (for example, when G and H are both unweighted graphs) scaling is less of an issue than when, for example, H is a weighted graph with a large dynamic range. Proper scaling of these matrices is an important area of research, and the discussion in Faust [2005] and references is a good starting point.

References

Nir Ailon and Bernard Chazelle. Approximate nearest neighbors and the fast johnson-lindenstrauss transform. In *Proceedings of the thirty-eighth annual ACM symposium on the Theory of computing*, pages 557–563, 2009.

Béla Bollobás. *Random Graphs*. Cambridge University Press, Cambridge, second edition, 2001.

Peter J. Carrington, John Scott, and Stanley Wasserman, editors. *Models and Methods in Social Network Analysis*. Cambridge University Press, Cambridge, UK, 2005.

Katherine Faust. *Using Correspondence Analysis for Joint Displays of Affilitation Networks*, pages 117–147. 2005.

Peter D. Hoff. Bilinear mixed-effects models for dyadic data. *Journal of the American Statistical Association*, 100(469):286–295, 2005.

Peter D. Hoff, Adrien E. Raftery, and Mark S. Handcock. Latent space approaches to social network analysis. *JASA*, 97:1090–1098, 2002.

Elizabeth Hohman and David J. Marchette. Hypergraph projection for text processing and information fusion. In *Proceedings of the Joint Statistical Meetings*, 2009.

Eric D. Kolaczyk. *Statistical Analysis of Network Data*. Springer, New York, 2009.

Zhiliang Ma and Carey E. Priebe. Supervised dimensionality reduction on the fusion of dissimilarity matrices. In *Proceedings of the Joint Statistical Meetings*, 2009.

David J. Marchette and Carey E. Priebe. Predicting unobserved links in incompletely observed networks. *Computational Statistics and Data Analysis*, 52:1373–1386, 2008.

Stanley Wasserman and Katherine Faust. *Social Network Analysis: Methods and Applications*. Cambridge University Press, Cambridge, UK, 1994.